Speed Daemon: Experience-based Mobile Robot Speed Scheduling

Chris J. Ostafew, Angela P. Schoellig, Timothy D. Barfoot
University of Toronto Institute for Aerospace Studies
Toronto, Ontario, Canada
chris.ostafew@mail.utoronto.ca, schoellig@utias.utoronto.ca,
and tim.barfoot@utoronto.ca

Jack Collier
Defence Research and Development Canada
Suffield, Alberta, Canada
jack.collier@drdc-rddc.gc.ca

***Abstract*—A time-optimal speed schedule results in a mobile robot driving along a planned path at or near the limits of the robot's capability. However, deriving models to predict the effect of increased speed can be very difficult. In this paper, we present a speed scheduler that uses previous experience, instead of complex models, to generate time-optimal speed schedules. The algorithm is designed for a vision-based, path-repeating mobile robot and uses experience to ensure reliable localization, low path-tracking errors, and realizable control inputs while maximizing the speed along the path. To our knowledge, this is the first speed scheduler to incorporate experience from previous path traversals in order to address system constraints. The proposed speed scheduler was tested in over 4 km of path traversals in outdoor terrain using a large Ackermann-steered robot travelling between 0.5 m/s and 2.0 m/s. The approach to speed scheduling is shown to generate fast speed schedules while remaining within the limits of the robot's capability.**

***Keywords*-Speed Scheduling; Experience-based; Mobile Robotics;**

I. Introduction

Trajectory planning through large-scale, unstructured environments is a challenging task for autonomous mobile robots. The goal is to generate feasible and safe trajectories despite effects from unknown terrain and unknown robot dynamics. In order to reduce the complexity of the problem, the problem is often divided into two subproblems: 1) path planning, where an algorithm identifies a safe path through a static environment, and 2) speed scheduling, where another algorithm plans the speeds at which the spatial path should be traversed [1]. Even once the path-planning problem is solved, calculating a time-optimal speed schedule is still a challenge. The main issue is in predicting the performance of robot subsystems, such as vision-based localization systems, as a function of variables such as speed and robot state in order to identify a constraint-satisfying speed schedule.

In this paper, we assume there exists a safe planned path. We present a speed scheduling algorithm that minimizes travel time and simultaneously guarantees feasibility of the trajectory despite unknown effects by incorporating experience from previous path traversals. For example, the speed scheduler uses localization experience to selectively increase the speed of sections of the path where localization reliability can be guaranteed. Motion blur at high speeds

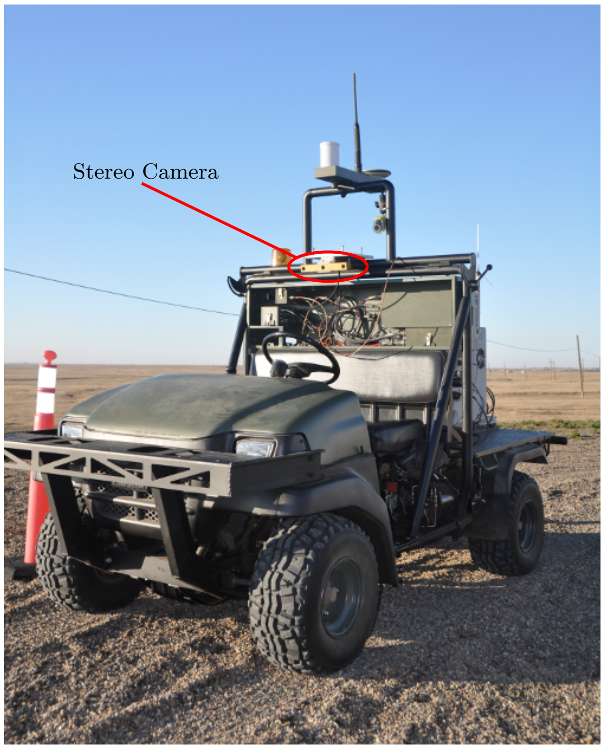

Figure 1. Experiments are performed using a large Defence Research and Development Canada (DRDC) Mule Research Vehicle (DMRV) where the sole sensor used for localization is a Point Grey Bumblebee stereo camera (highlighted in red). The experience-based speed scheduler presented in this paper is able to identify speed limits that maintain the reliability of the vision-based localization system.

is one cause of reduced localization reliability and is very difficult to predict *a priori*. To our knowledge, this is the first algorithm to incorporate experience from previous path traversals when producing a speed schedule.

There are many speed scheduling approaches in the literature. For schedulers seeking minimum-time trajectories, the general approach is to identify limits on the robot speed and acceleration as a function of constraints, such as actuator limits, then plan a schedule to operate at the highest speed possible. As a result, most approaches differ primarily in which constraints are addressed and at what point in the planning process they are included.

Classic speed schedulers for mobile robots concentrate on generating smooth speed profiles prior to commencing path traversal while incorporating electromechanical constraints on speeds and accelerations [2, 3]. For example, Munoz et al. [4] derive the maximum robot speed, acceleration, and deceleration from motor and brake system specifications. Prado et al. [5, 6] include speed and acceleration limits resulting from predicted motor temperature and battery power. Other schedulers predict lateral accelerations, friction forces, and weight transfers to constrain the speed based on slip and path-tracking error limits [5, 7, 8]. In each of these papers, detailed models of the robot and environment are required to generate the appropriate speed or acceleration limits. In this paper, in addition to simple *a priori* speed and acceleration limits, we use past observations of vision-system performance, path-tracking errors, and control inputs to iteratively identify speed and acceleration limits.

Other approaches schedule the speed while tracking a path, using experiences in real-time to identify speed limits. For example, the Stanley robot, designed for the 2006 DARPA Grand Challenge, slows down after encountering rough sections along a planned path [9]. Path roughness is identified using measurements of the vertical acceleration of the vehicle [10]. The approach is shown to decrease the damage to the robot caused by shock and vibration, thereby increasing the long-term system reliability. While the approach employed by Thrun et al. [9] is designed for systems traversing a path for the first time, it nevertheless results in the robot experiencing serious shock and vibrations at the start of rough patches along a path. Our approach, on the other hand, schedules speed based on experience from previous path traversals. This gives our system the ability to slow down *before* encountering challenging sections of the path, rather than behaving reactively. Finally, in some speed schedulers, the speed is also determined in real-time in order to prevent collisions with dynamic obstacles [1, 5, 7]. In these cases, it is assumed that the dynamic obstacle is travelling across the path and therefore waiting a few moments will result in a clear path. An investigation into the sensing, control, and actuation requirements for real-time obstacle avoidance is presented by Kelly and Stentz [11, 12]. In this paper, we assume the environment is free of dynamic obstacles and the planned path avoids static obstacles.

In summary, the key contribution of this paper is a speed scheduling algorithm that incorporates both *a priori* speed and acceleration limits, and experience-based constraints. The algorithm starts with a conservative speed schedule, then over sequential path traversals the algorithm increases the speed in sections of the path where it is feasible. By incorporating experience, we are able to address limits on speed arising from complex vision-based localization systems and path-tracking controllers in challenging off-road terrain.

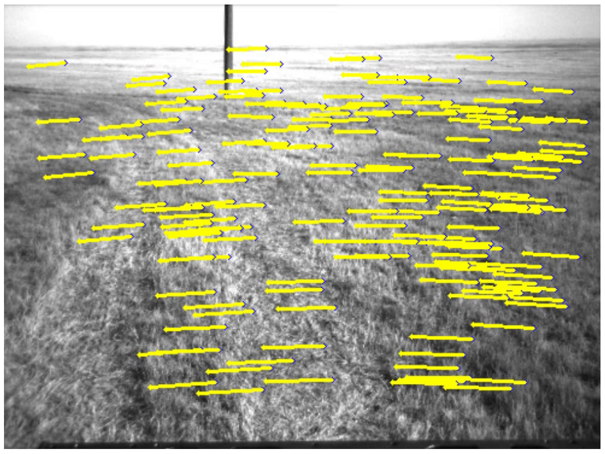
Figure 2. A visual representation of re-localization in our VT&R framework. Each feature track represents the translation between a feature identified during the teach phase and re-identified during a repeat phase.

II. Visual Teach and Repeat

Localization for the mobile robot used in this paper is provided by an on-board Visual Teach & Repeat (VT&R) algorithm developed by Furgale and Barfoot [13] where the sole sensor is an on-board stereo camera (see Fig. 1). In the first operational phase, the teach phase, the robot is driven along the desired path manually by an operator. Localization in this initial operation is obtained relative to the robot's starting position by visual odometry (VO). In addition to the VO pipeline, path vertices are defined along the path by storing keyframes composed of local feature descriptors and their 3D positions. During the second operational phase, the repeat phase, the robot re-localizes against the stored keyframes, generating an estimate of the pose of the robot relative to the nearest path vertex. Re-localization depends on matching features and can be visualized by feature tracks between the current robot view and the teach-pass robot view (Fig. 2). As long as sufficient correct feature matches are made, the system generates consistent localization and has the ability to support an iterative speed scheduling scheme.

III. Speed Planning

A. Path-Tracking Control

The mobile robot path-tracking controller follows a trajectory consisting of a set of N desired poses (Fig. 3),

$$\mathcal{P}_d := \{\mathbf{x}_{d,i} \,|\, i \in 1 \ldots N\},$$

each defining a translation (x, y, z) and rotation (roll, pitch, yaw), $\mathbf{x}_{d,i} := (x, y, z, r, p, y)$, relative to the origin of the path, and N scheduled speeds,

$$\mathcal{V}^j_{\text{sched}} := \{v^j_{\text{sched},i} \,|\, v \in \mathbb{R},\, i \in 1 \ldots N\},$$

generated for the jth trial by the speed scheduler defined in Sec. III-B. At a given time k, the VT&R algorithm produces an estimate of the robot's pose, $\mathbf{x}_k$, relative to the

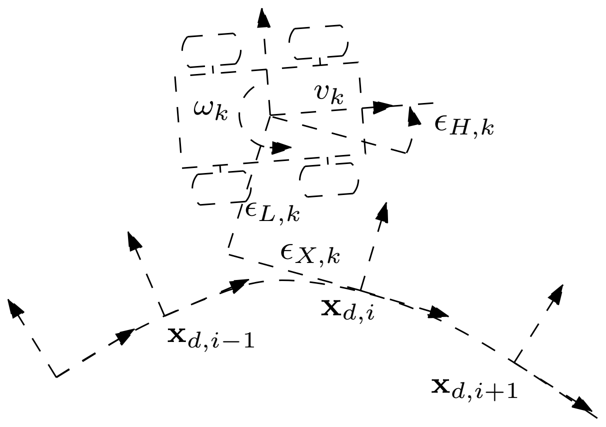

Figure 3. Definitions of the robot velocities, v_k and ω_k, and three path-tracking errors, $\epsilon_{X,k}$, $\epsilon_{L,k}$ and $\epsilon_{H,k}$, defined relative to the nearest path vertex by Euclidean distance. The role of the speed scheduler is to assign linear speeds, $v_{\text{sched},i}$, for the robot to use while in the neighbourhood of each path vertex.

nearest path vertex by Euclidean distance. The path-tracking controller then sets the commanded linear velocity of the robot, $v_{\text{cmd},k}$, based on the scheduled speed at the nearest path vertex, $v_{\text{sched},i}$, and computes a commanded angular velocity, $\omega_{\text{cmd},k}$, using feedback control. The controller aims to minimize the heading and lateral path-tracking errors (Fig. 3), $\epsilon_{H,k}$ and $\epsilon_{L,k}$, respectively, using feedback linearization [14].

B. Speed-Scheduler Algorithm Overview

The automated speed scheduler proceeds in several steps:

1) Initiate the first schedule, $\mathcal{V}_d^1$, with a safe speed
2) Travel the path and collect experience:
   - Vision-based experience
   - Path-tracking experience
   - Control-input experience
3) Suggest a speed schedule for the next trial based on experience
4) Limit the suggested speed schedule based on *a priori* speed and acceleration constraints
5) Repeat 2) - 4)

C. Collected Experience

During each trial, the robot drives the full path and accumulates experience. Specifically, we collect vision-based localization, path-tracking, and control-input experience for use in speed scheduling. Here we introduce k_i as the time index at which the ith vertex is passed.

1) Vision-based Localization Experience: When using vision-based localization systems, there exists a speed limit above which localization becomes unreliable and the safety of the robot can no longer be assured. This speed limit may come as a result of low light conditions, a degraded scene (relative to when the path was taught), large deviations from the path, or perhaps motion blur. Instead of trying to predict the effect of these conditions, we record the number of features matched by the VT&R system, m_i^j, when passing the ith vertex during the jth trial as an indicator of the conditions faced by the localization system,

$$\mathcal{M}^j := \{m_i^j \,|\, i \in 1 \ldots N\}.$$

Intuitively, we make the assumption that there exists a relationship such that as the speed of the robot increases, the number of matched features at a given path vertex decreases. While this relationship is not known *a priori*, we use experience to judge whether the speed at the ith path vertex during the next trial can be increased further.

2) Path-tracking Experience: We also record the lateral and heading path-tracking errors (Fig. 3), $\epsilon_{L,i}^j$ and $\epsilon_{H,i}^j$, respectively,

$$\begin{bmatrix} \epsilon_{L,i}^j \\ \epsilon_{H,i}^j \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{k_i}, \qquad (1)$$

when passing the ith vertex during the jth trial,

$$\boldsymbol{\epsilon}_L^j := \{\epsilon_{L,i}^j \,|\, i \in 1 \ldots N\},$$
$$\boldsymbol{\epsilon}_H^j := \{\epsilon_{H,i}^j \,|\, i \in 1 \ldots N\}.$$

We do this for two reasons. First, we have assumed that the planned path is safe and free of obstacles (Sec. I). Therefore, it is important to maintain low path-tracking errors since we can only guarantee that the terrain near the planned path is free of obstacles. In the case that the path planner provides additional information about the lateral distance to obstacles, our speed scheduler could restrict the robot speed so as to ensure sufficiently low lateral path-tracking errors. Second, the vision system is sensitive to perspective changes between the teach pass and any repeat pass. Perspective changes are the direct result of path-tracking errors and reduce the reliability of the localization system. Prediction of path-tracking errors as a function of speed is challenging. Thus we use experience to judge the effect of speed on path-tracking errors.

3) Control-input Experience: Finally, the scheduled linear speed must also address constraints on angular velocities resulting from actuator limits. As a result, we record the commanded angular velocity, $\omega_{\text{cmd},i}^j = \omega_{\text{cmd},k_i}$, during the jth trial when passing the ith path vertex,

$$\Omega^j := \{\omega_{\text{cmd},i}^j \,|\, i \in 1 \ldots N\}.$$

In order to track the desired path at a velocity, v_k, the robot must be capable of turning at an angular velocity, ω_k, as shown in Fig. 3. In order to achieve this actual angular velocity, the robot path-tracking controller commands a commanded angular velocity, $\omega_{\text{cmd},k}$, compensating for wheel slip, side slopes, and other model discrepancies through state feedback. As the commanded linear velocity is increased over trials, so too the commanded angular velocity will increase in order to track the path. In practice, detailed

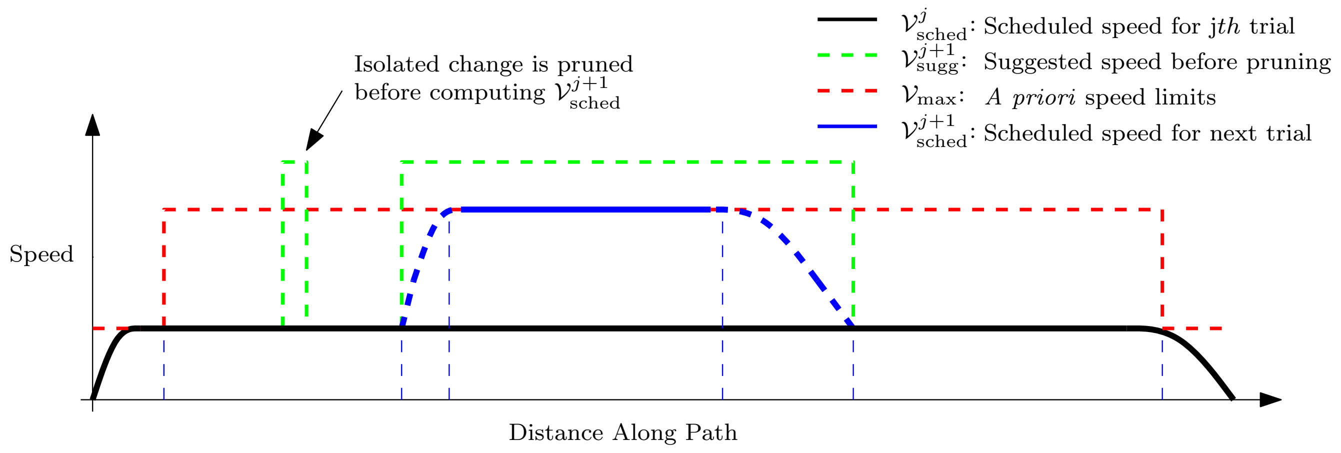

Figure 4. Experience-based speed scheduling occurs in two steps: 1) modifications to the previous speed schedule are suggested based on experience (Green), 2) the suggested speeds are constrained by *a priori* speed and acceleration constraints (Blue).

models of wheel slip including the effects of side-slopes and ground texture are generally not available for every situation. For example, when a robot is required to traverse a straight path across a sideslope, wheel-ground interaction models are still under development [15]. However, by using experience, we are able to predict that the commanded angular velocities remain within the actuator limits when generating new speed schedules.

D. Experience-based Speed Schedule Modification

The next step in the speed-scheduler algorithm (Step 3) is to suggest new desired speeds for the next trial for each path vertex based on the desired speeds during the jth trial and the collected experience (Fig. 4). Using tuned values for increasing and decreasing the scheduled speed, $\gamma_1 > 0$, $\gamma_2 > 0$, respectively, and thresholds, $\lambda_L \geq 0$, $\lambda_H \geq 0$, $\lambda_\omega > 0$, and $\lambda_{\text{feat}} \geq 4$, the scheduler follows rules to generate the suggested speeds for each path vertex:

$$v^{j+1}_{\text{sugg},i} = \begin{cases} v^{j}_{\text{sched},i} + \gamma_1 & \text{if } (|\epsilon^{j}_{L,i}| < \lambda_L) \wedge (|\epsilon^{j}_{H,i}| < \lambda_H) \wedge \\ & \quad (|\omega^{j}_{\text{cmd},i}| < \lambda_\omega) \wedge (m^{j}_{i} > \lambda_{\text{feat}}), \\ v^{j}_{\text{sched},i} - \gamma_2 & \text{if } (|\epsilon^{j}_{L,i}| > \lambda_L \lambda_{db}) \vee (|\epsilon^{j}_{H,i}| > \lambda_H \lambda_{db}) \vee \\ & \quad (|\omega^{j}_{\text{cmd},i}| > \lambda_\omega \lambda_{\text{db}}) \vee (m^{j}_{i} < \lambda_{\text{feat}}/\lambda_{\text{db}}), \\ v^{j}_{\text{sched},i} & \text{otherwise.} \end{cases} \tag{2}$$

We use $\lambda_{\text{db}} > 1$ to produce a deadband where the speed at a vertex is neither increased nor decreased. After generating suggested speeds for all path vertices in the next trial, isolated increases are then pruned, $v^{j+1}_{\text{sugg,i}} \leftarrow v^{j}_{\text{sched},i}$, to encourage a smooth speed profile (Fig. 4). An isolated increase occurs when too few path vertices in a section of the path are eligible for increased speeds. Finally, for the first trial, when there is no experience from which to draw, the scheduled speed for all path vertices is set to a fixed speed, $v^{1}_{\text{sched},i} = v_{\text{init}}$.

E. A Priori Speed and Acceleration Constraints

The final step in our experience-based speed scheduler (Step 4) is to incorporate several *a priori* constraints on linear speed, acceleration, and deceleration. Firstly, the robot has a known actuator-based linear speed constraint. In addition, the robot must respect speed limits for safety, particularly during key sections of the path such as the start and end of the path. The combination of actuator and safety constraints results in a set of speed limits,

$$\mathcal{V}_{\text{max}} := \{v_{\text{max},i} \,|\, i \in 1 \ldots N\},$$

as shown in Fig. 4. These speed limits are applied to the suggested scheduled speed,

$$v^{j+1}_{\text{sched},i} \leftarrow \min\{v^{j+1}_{\text{sugg},i}, v_{\text{max},i}\}. \tag{3}$$

Finally, we limit the acceleration and deceleration to account for robot capability and safe operation. The scheduled speed must satisfy the acceleration and deceleration constraints, $a_{\text{max}} > 0$ and $a_{\text{min}} < 0$, respectively, at every path vertex,

$$v^{j+1}_{\text{sched},i} \leftarrow \min\Big\{ v^{j+1}_{\text{sched},i}, \sqrt{\left(v^{j+1}_{\text{sched},i-1}\right)^2 + 2\, d_{i-1,i}\, a_{\text{max}}}, \sqrt{\left(v^{j+1}_{\text{sched},i+1}\right)^2 - 2\, d_{i,i+1}\, a_{\text{min}}} \Big\}, \tag{4}$$

where $d_{i,j}$ is the distance between two path vertices, $\mathbf{x}_{d,i}$ and $\mathbf{x}_{d,j}$, in the ground plane (neglecting the z component),

$$d_{i,j} = \sqrt{(x_{d,j} - x_{d,i})^2 + (y_{d,j} - y_{d,i})^2}. \tag{5}$$

The resulting speed schedule satisfies all *a priori* speed and acceleration constraints, and also takes into account localization performance requirements and path-tracking performance requirements.

IV. Experimental Results

A. Overview

The speed-scheduling algorithm presented in Sec. III was tested in two experiments with 10 trials each, resulting in over 4 km of testing in outdoor environments. The test vehicle (Fig. 1) was manually taught the two paths including sharp turns, a variety of slopes, and a variety of surfaces. Both paths were taught at the DRDC Experimental Proving Grounds in Suffield, Alberta, Canada. The resulting speed schedules varied in speed from 0.5 m/s to 2.0 m/s.

B. Tuning Parameters

The speed scheduler was set to maintain matched feature counts greater than 30, heading errors less than $10°$, lateral errors less than 15 cm, and commanded angular velocities less than 1.0 rad/s. The speed scheduler increments, γ_1 and γ_2, were set to increase a scheduled speed by 0.2 m/s or decrease a scheduled speed by 0.24 m/s. The vehicle speed was limited to 0.4 m/s during the start and end segments, and 2.0 m/s otherwise. Finally, the vehicle acceleration was limited to 0.2 m/s^2 and the vehicle deceleration was limited to -0.05 m/s^2. Setting a slow deceleration limit ensures the vehicle begins to slow down far in advance of slow speed sections.

C. Results

During the first experiment, the vehicle travelled the desired 100-m-long path (Figs. 5 and 6) 10 times, resulting in 1 km of testing. During the first four trials, the number of matched features, path-tracking errors, and desired angular speeds were all within the specified limits. As a result, the scheduled speed was increased equally for all path vertices except those near the start and end of the path (Figs. 7 and 8). In the fifth trial, the path-tracking error began limiting the scheduled speed at 50 m along the path (Fig. 9). In general, the path-tracking errors coincided with the path slopes and curvatures (Figs. 5 and 6). Finally, as the speed of the vehicle increased, the number of matched features decreased (Fig. 9). However, the reduction in matched features was not high enough to limit the speed of the vehicle.

During the second experiment, the vehicle travelled a 375-m-long path 10 times, resulting in over 3 km of testing (Figs. 10 and 11). As in the first experiment, the number of matched features, path-tracking errors, and commanded angular speeds were all within the specified limits for the first few trials resulting in a rapid increase in scheduled speeds (Fig. 10). In this case, we were able to initialize the vehicle at a relatively safe speed of 0.5 m/s and the speed scheduling algorithm was able to autonomously reduce the travel time significantly (Fig. 11).

D. Discussion and Future Work

The algorithm proved to make appropriate decisions with respect to the localization system and path-tracking errors.

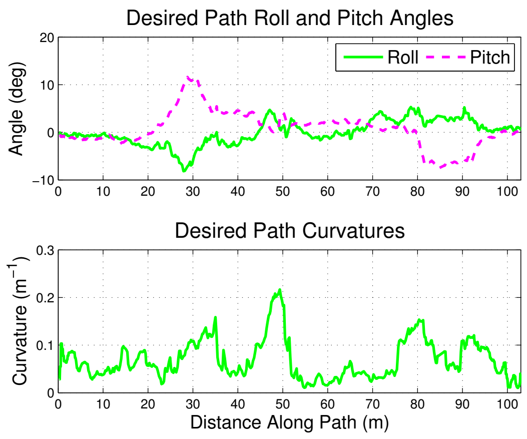

Figure 5. Experiment 1 Path Conditions: The test path for the first experiment included slope angles up to $10°$, side-slope angles up to $10°$, and path curvatures up to $0.2\ \mathrm{m}^{-1}$.

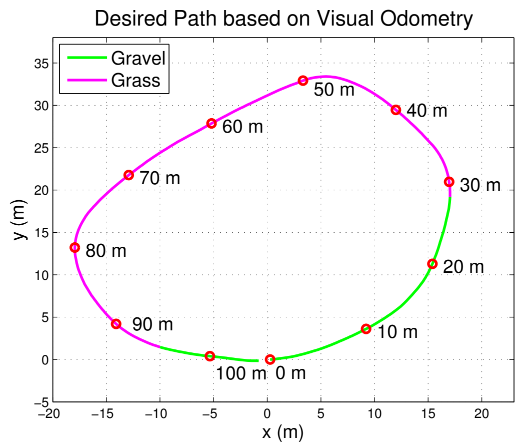

Figure 6. Experiment 1 Test Route: The path for the first experiment was approximately 100 m long and included gravel and grassy terrain.

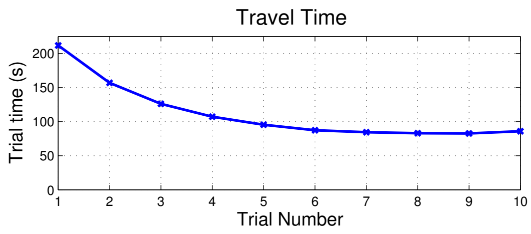

Figure 7. Experiment 1 Travel Time vs Trial: The strategic increases in scheduled speed (Fig. 8) resulted in a significantly reduced travel time by the 10th trial. Once the speed schedule had converged, variations in the travel time were due largely to non-repeatable disturbances affecting the vehicle speed and path-tracking errors.

In retrospect, however, it may have been possible to estimate the available speed increase more accurately by regressing from experience, including more than just the previous trial. This may have increased the speed increments during the early trials. Furthermore, it may be useful to maintain experience as a function of time as well. For example, it is quite common for vision-based localization systems

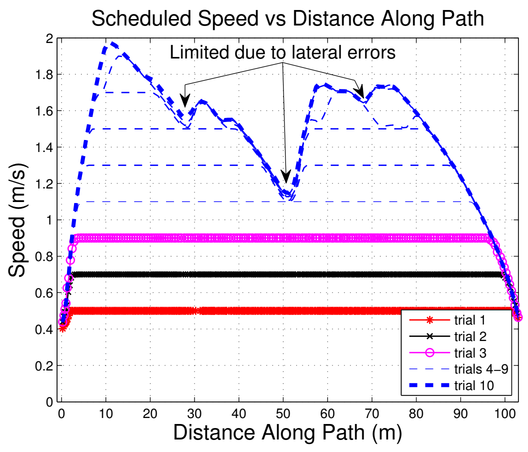

Figure 8. Experiment 1 Scheduled Speed vs Distance: The speed scheduler maximizes speed while taking into account limits derived from the vision system, path-tracking errors, and control inputs (Fig. 9). During the 10th trial, the slowest scheduled speed occurs at around 50 m when the lateral path-tracking error is largest.

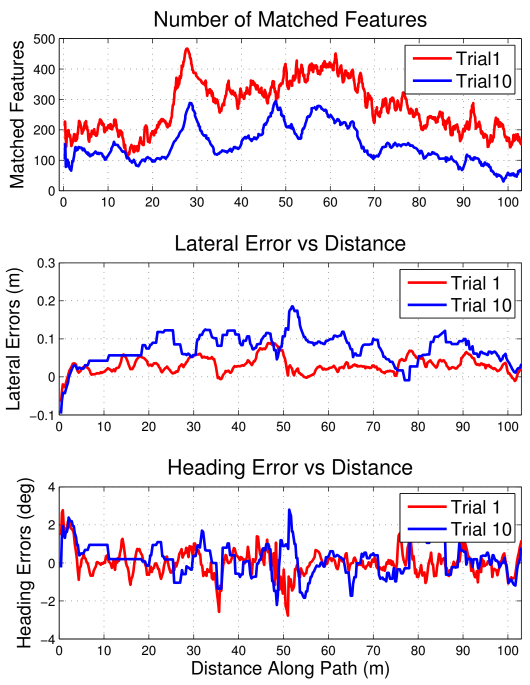

Figure 9. Experiment 1 Matched Features and Path-tracking Errors vs Distance: The number of matched features in trial 10 were reduced due to motion blur (Fig. 8) and variations in lighting relative to the first trial. Heading errors also did not affect the scheduled speed. On the other hand, increased speed resulted in increased lateral path-tracking errors, which ended up being the most common cause of limited speed.

operating outdoors to experience periodic lighting changes affecting the reliability of the system to localize. In such a case, one could imagine the system *anticipating* a poorly lit section of a path as a function of time-of-day and slowing down in advance. This may be especially beneficial for a

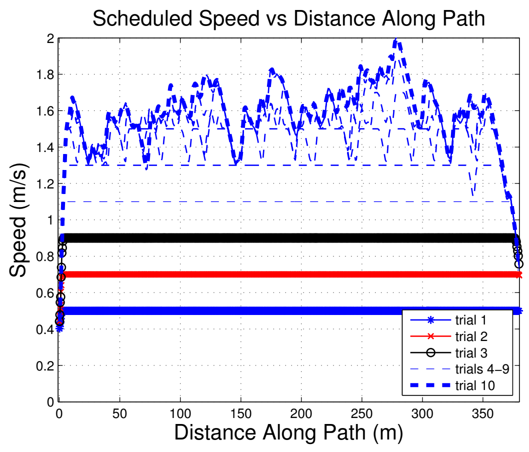

Figure 10. Experiment 2 Scheduled Speed vs Distance: During the second experiment, the experience-based speed scheduler generated a profile ranging from 0.5 m/s to 2.0 m/s after 10 trials, resulting in a significant reduction in travel time (Fig. 11).

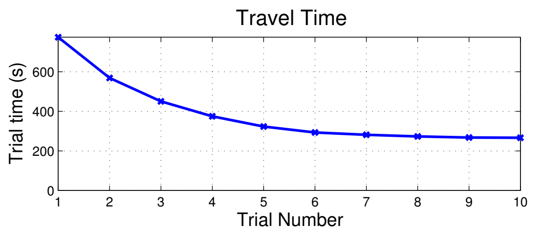

Figure 11. Experiment 2 Travel Time vs Trial: During the first four trials, the number of matched features, path-tracking errors, and commanded angular speeds were all within the specified limits resulting in a significant decrease in travel time.

system that does not repeat a path frequently enough to see the lighting change gradually. Finally, it may also be of interest to address the possibility of dynamic obstacles in general, and possible recurring dynamic obstacles. Dynamic obstacles are generally addressed in real-time. However, in some cases dynamic obstacles cross paths with predictable timing.

V. Conclusion

In summary, this paper presents an experience-based speed scheduler for path-repeating mobile robots. The algorithm collects localization, path-tracking, and control-input experience then generates a speed schedule for the next trial in two broad steps. First, the algorithm suggests a new speed profile based on the collected experience and the previous speed schedule. Second, the alogorithm limits the suggested profile based on *a priori* speed and acceleration limits. The algorithm was implemented and tested on a large DMRV (Fig. 1) in two experiments with 10 trials each, resulting in over 4 km of driving. The algorithm proved to be effective at maximizing the robot speed while taking into account limits that are very difficult to predict in advance, such as those imposed by vision-based localization systems.

Acknowledgment

This research was funded by the Ontario Ministry of Research and Innovation's Early Researcher Award Program and the Natural Sciences and Engineering Research Council (NSERC) through the NSERC Canadian Field Robotics Network (NCFRN).

References

[1] K. Kant and S. W. Zucker. Toward efficient trajectory planning: The path-velocity decomposition. *The International Journal of Robotics Research*, 5(3):72–89, 1986.

[2] R. Fatouhi, W. Szyszkowski, and P. Nikiforuk. Trajectory planning and speed control for a two-link rigid manipulator. *Journal of Mechanical Design*, 124(3): 585–589, 2002.

[3] C.G.L. Bianco. Kinematically constrained smooth real-time velocity planning for robotics applications. In *Proceedings of the IEEE International Conference on Control and Automation*, pages 373–378, 2009.

[4] V. Munoz, A. Ollero, M. Prado, and A. Simon. Mobile robot trajectory planning with dynamic and kinematic constraints. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2802–2807, 1994.

[5] M. Prado, A. Simón, E. Carabias, A. Perez, and F. Ezquerro. Optimal velocity planning of wheeled mobile robots on specific paths in static and dynamic environments. *Journal of Robotic Systems*, 20(12):737–754, 2003.

[6] M. Prado, A. Simón, and F. Ezquerro. Velocity, acceleration and deceleration bounds for a time-optimal planner of a wheeled mobile robot. *Robotica*, 20(2): 181–193, 2002.

[7] O. Purwin and R. D'Andrea. Trajectory generation and control for four wheeled omnidirectional vehicles. *Robotics and Autonomous Systems*, 54(1):13–22, 2006.

[8] I. Waheed and R. Fotouhi. Trajectory and temporal planning of a wheeled mobile robot on an uneven surface. *Robotica*, 27(4):481–498, 2009.

[9] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, et al. Stanley: The robot that won the DARPA grand challenge. *Journal of Field Robotics*, 23(9):661–692, 2006.

[10] C. A. Brooks and K. Iagnemma. Vibration-based terrain classification for planetary exploration rovers. *IEEE Transactions on Robotics*, 21(6):1185–1191, 2005.

[11] A. Kelly and A. Stentz. Rough terrain autonomous mobility (part 1): A theoretical analysis of requirements. *Autonomous Robots*, 5(2):129–161, 1998.

[12] A. Kelly and A. Stentz. Rough terrain autonomous mobility (part 2): An active vision, predictive control approach. *Autonomous Robots*, 5(2):163–198, 1998.

[13] P. Furgale and T. Barfoot. Visual teach and repeat for long-range rover localization. *Journal of Field Robotics*, 27(5):534–560, 2010.

[14] C. Samson and K. Ait-Abderrahim. Feedback control of a nonholonomic wheeled cart in cartesian space. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 1136–1141, 1991.

[15] K. Nagatani, T. Noyori, and K. Yoshida. Development of multi-d.o.f. tracked vehicle to traverse weak slope and climb up rough slope. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 2849–2854, 2013.